参数自适应的网格密度聚类算法 *

郑 诚[a, b]，曹 杨[a, b]

(安徽大学 a. 计算智能与信号处理教育部重点实验室; b. 计算机科学与技术学院，合肥 230601)

摘 要： 针对网格密度聚类算法存在的网格宽度和密度阈值难以确定、以及聚类精度不高的缺陷，提出了一种参数自适应的网格密度聚类算法。定义了数据集的标准化离散度的概念，运用数据集的自然分布信息自适应的计算出每一维较优的分割宽度，对不同的密度阈值统计其噪声样本对象的数量，绘制了噪声曲线，从噪声曲线中获得最佳的密度阈值，而且增加了类簇边缘处理技术，进一步提高了聚类的质量。仿真实验表明，改进后的算法可获得更好的聚类效果。

关键词： 网格密度；聚类；空间划分；噪声曲线

中图分类号： TP301.6 doi: 10.3969/j.issn.1001-3695.2018.04.0289

Self-adaptive based on grid density clustering algorithm

Zheng Cheng[a, b], Cao Yang[a, b]

(*a. Key Laboratory of Intelligent Computing & Signal Processing of Ministry of Education, b. College of Computer Science & Technology Anhui University, Hefei* 230601, *China*)

Abstract: The clustering algorithm of based on grid density is difficult to determine the grid width and density threshold. In addition, the accuracy of the results is dissatisfied. Considering the problem above, this paper proposed an improved clustering algorithm. The better segmentation width of each dimension is calculated by the natural distribution information of the data set. According to the different density thresholds, the number of the noise is calculated. The noise curve is drawn. The best density threshold is obtained from the noise curve. Simulation results show that the improved algorithm can get better clustering results.

Key words: grid density; clustering; space partition; noise curve

0 引言

聚类分析是数据挖掘领域中常用的技术手段，常见的聚类算法有基于划分的聚类算法[1~4]，基于层次的聚类算法[5~8]，基于密度的聚类算法[9~11]，基于网格的聚类算法[12~14]，基于模型的聚类算法[15,16]和基于网格密度的聚类算法[17,18]。除此之外，一些新颖的聚类算法被国内外的学者所提出，基于马尔可夫随机游走的谱聚类算法[19]，基于确定性退火的聚类算法[20]，以及将智能优化算法与聚类相结合的算法[21]。不同类型的聚类算法有各自的优缺点，目前为止，没有一种聚类算法能对所有的数据集都有很好的聚类效果。

基于网格密度的聚类算法是一种常见的聚类算法。该算法既具有密度聚类算法发现任意形状类簇的优势，又具有网格聚类算法高效的特点。此外，该算法对于噪声数据不敏感，具有很好的去噪能力;并且，网格密度聚类算法具有很强的伸缩能力，因此，该算法非常适合对大规模数据集的聚类操作。基于网格密度的聚类算法虽然具有诸多的优点，但是，该算法同样需要两个输入参数（每一维划分的单元格数目和识别稠密单元格的密度阈值）作为算法的基础，然而，参数的设置不同，将会对聚类结果产生很大的影响。此外，运用网格密度聚类算法进行聚类操作，得到的类簇边缘呈直线型，这就导致了该算法的聚类精度不是太高。为此，本文对网格密度聚类算法进行了改进。本文的主要创新点是：a）运用数据集的内部分布信息来自动的确定每一维的划分数目;b）构造了数据集的噪声曲线，运用噪声曲线来确定合理的密度阈值；c）增加了对稀疏单元格的处理技术，从而达到了提高聚类精度的效果。

1 网格密度聚类算法的介绍及其缺陷分析

基于网格密度的聚类算法（clustering algorithm of based on grid density，BGD）是以网格为最小的处理单元，从而大大提高了聚类的速度。

1.1 基本概念

定义 1 数据集 S 的划分。将数据集 S 所分布空间的每一维划分为相等的间隔段，生成不相交的矩形或者超矩形单元集

收稿日期： 2018-04-25；**修回日期：** 2018-06-13 **基金项目：** 安徽省高校自然科学研究重点项目（KJ2013A020）

作者简介： 郑诚（1964-），男，安徽合肥人，副教授，博士，主要研究方向为数据挖掘、智能语义信息检索（1290672922@qq.com）；曹杨（1996-），女，安徽合肥人，硕士研究生，主要研究方向为数据挖掘、聚类.

合 G，集合 G 覆盖整个数据集 S 的数据分布空间。集合 G 中的每一个矩形或者超矩形单元 $grid$ 的空间位置表示为 $\{c_1,c_2,c_3,\ldots,c_d\}$，其中 $c_i=[l_i,h_i)$ 对应于第 i 维的一个左闭右开的间隔段。一个单元还可以表示为（$cNum_1$，$cNum_2$，…，$cNum_d$），其中 $cNum_i$ 是区间 $[l_i,h_i)$ 对应的间隔序号，每一维的间隔序号的编号从 1 开始。

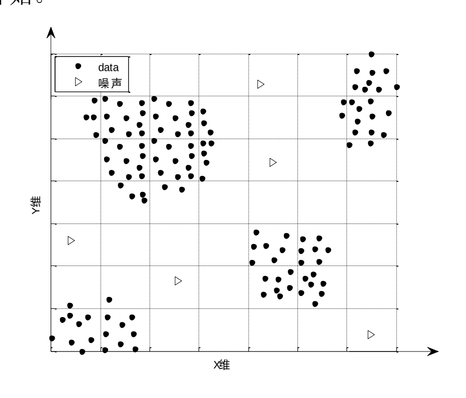


图 1　数据集 dataset1 的划分

图 1 就是二维数据集 dataset1 的一种划分，图 1 中二维数据集的每一维都被划分为 7 个间隔段。

定义 2　数据对象 x_i 的映射。数据对象 x_i 的映射就是根据式（1）将 x_i 映射到对应的矩形或者超矩形单元 $grid$ 中。

$$cNum_j=\frac{x_{ij}-min_j}{\frac{max_j-min_j}{m}}=\frac{m\left(x_{ij}-min_j\right)}{max_j-min_j}\qquad(1)$$

其中：x_{ij} 是数据对象 x_i 的第 j 维属性值。m 是数据集每一维的划分段数目（以下简称：维度分割参数）。min_j，max_j 分别为数据集 S 中第 j 维属性值的最小值和最大值。

定义 3　单元格 $grid$ 的密度 den（$grid$）。单元格 $grid_k$ 的密度就是数据集被划分后，映射到该矩形或者超矩形单元格 $grid$ 中数据对象的个数。

定义 4　稠密单元格。稠密的单元格就是单元格的密度大于等于某一给定的密度阈值：

$$den\left(grid\right)\geq\delta_0\qquad(2)$$

其中：δ_0 是密度阈值，由用户根据经验设定。

定义 5　稀疏单元格。稀疏的单元格和稠密的单元格相反，密度小于给定的密度阈值：

$$den\left(grid\right)<\delta_0\qquad(3)$$

其中的 δ_0 和式（2）中的一样。

定义 6　单元格 $grid_i$ 和 $grid_j$ 是连通的。单元格 $grid_i$ 和 $grid_j$ 是连通的，当且仅当 $grid_i$ 和 $grid_j$ 有一个公共面（高维情况是超面），或者 $grid_i$ 和 $grid_j$ 都与第三个单元格 $grid_k$ 有一个公共面。

1.2　BGD 算法的思想流程

网格密度聚类算法的主要思想就是划分数据空间，寻找稠密网格单元格的最大连通区域，得到的每一个最大连通区域中的数据对象，组成了数据集的一个类簇。BGD 算法的流程如下：

a)根据维度分割参数 m 对数据集 S 的数据空间进行划分，得到网格单元集合 $G=\{grid_1，grid_2，grid_3，…，grid_w\}$。

b)通过式（1）将数据集 S 中的每一个数据对象 xi 映射到对应的网格单元中。

c)根据密度阈值参数 $\delta0$，标记稠密网格单元和稀疏网格单元。

d)从网格单元集合 G 中选择一个稠密网格单元，以此稠密网格单元为起点，寻找稠密网格单元的最大连通区域 q_i，将寻找到的稠密连通单元格从集合 G 中删除。

e)判断网格单元集合 G 中是否还存在稠密的网格单元，若存在，重复执行 Step4，得到稠密网格单元的最大连通区域集合 $Q=\{q_1，q_2，…，q_k\}$。

f)将的得到的每一个稠密单元格的最大连通集合 q_i 中的数据对象输出，每一个最大连通集合中的数据即为一个类簇。

1.3　BGD 算法存在的缺陷分析

网格密度聚类算法虽然具有诸多的优点，但是，该算法的缺点也是显而易见的。首先，该算法需要维度分割参数 m 和密度阈值参数 δ_0 作为它的基础，而这两个参数对于聚类结果往往有很大的影响。例如，若每一维的分割参数 m 的取值过大，很可能将属于不同类簇的数据对象划分到同一个网格单元中；若分割参数 m 的取值过小，就起不到加快聚类速度的效果。对于密度阈值 δ_0 的选取，对聚类结果的影响更大，为了说明这个问题，我们用图 1 中的数据划分为例。图 1 中，如果密度阈值 δ_0 的取值不同，则聚类的结果将会有很大的不同，如图 2 所示。

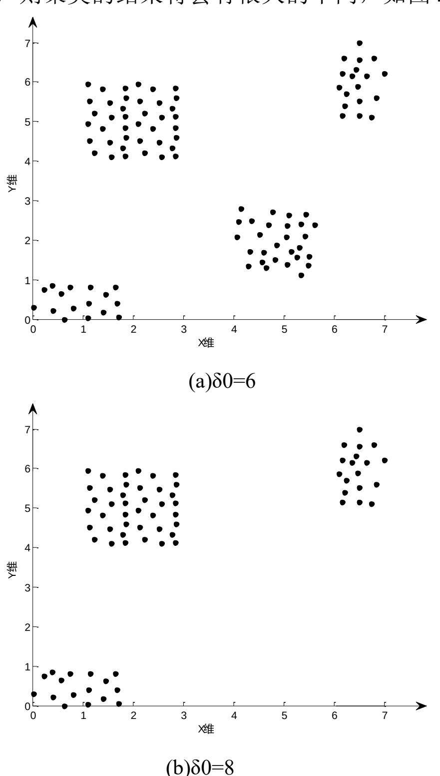


图 2　聚类结果

图 2（a）是密度阈值 $\delta_0=6$ 时数据集 dataset1 的聚类结果，从图 2（a）可以看出，当密度阈值 $\delta_0=6$ 时，聚类结果是理想

的，可以把数据集的四个自然簇都显示出来，并且把噪声数据排除在外。但是，如图 2（b）所示，当密度阈值 δ_0=8 时，右下角类簇的密度相对较小，划分后，被识别为稀疏的单元格，最后被认为是噪声而被抛弃，从而导致聚类结果非常差，只识别出了数据集中的其它三个类簇。另外，从图 2（a）（b）中还可以看出，类簇的边缘都是被垂直或者水平切割的，聚类结果失去了自然类簇边缘的平滑度，这是因为类簇的边缘被误认为是噪声数据而被丢弃，这也是导致网格密度聚类算法聚类精度不高的重要原因之一。

2　参数自适应的网格密度聚类算法

针对 2.3 节中所述的网格密度聚类算法存在的缺陷，本节提出了参数自适应的网格密度聚类算法（self-adaptive based on grid density clustering algorithm，SA-BGD）。以下分别介绍 SA-BGD 算法确定维度分割参数 *m* 值的方案、密度阈值 *δ0* 的选取策略以及类簇边缘的处理技术。

2.1　维度分割参数 m 的确定

为了得到合理的分割参数 *m*，SA-BGD 算法根据待聚类数据集自身的特点来计算维度分割参数 *m*。为便于后面的叙述，进行以下新的定义。

定义 7　第 *j* 维数据的离散度 D_j。Dj 是数据集第 *j* 维数据的离散度，可由式（4）来计算。

$$D_j = \frac{S_j}{X_means_j} \quad (4)$$

其中：S_j 和 X_means_j 分别是第 *j* 维数据的标准差和均值。

定义 8　数据集的标准化离散度 D_s。D_s 是数据集的标准化离散度，其形式化定义如式（5）所示。

$$D_s = \frac{1}{\left(1+\frac{1}{d}*\sum_{j=1}^{d} D_j\right)} \quad (5)$$

其中：*d* 是数据集中数据对象的维数。D_s 的值越小，数据集的离散度越大；D_s 的值越大，数据集的离散度越小。

数据集的维度分割参数 *m* 的值可由式（6）来计算。

$$m = D_s * \sqrt[d]{N} \quad (6)$$

其中：*d* 是数据集中数据对象的维数，*N* 是数据集中样本的数量。运用式（6）计算的分割参数 *m* 是合理的，因为它充分考虑了数据集中数据的分布情况。

2.2　密度阈值 δ_0 的选取

为了得到合理的密度阈值 δ_0，首先利用式（6）计算维度分割参数 *m* 的值，用 *m* 来分割数据空间，并且将数据对象映射到对应的网格单元中，统计出每个网格单元中的数据对象的个数，保存在各自的网格单元中。接下来，分别统计出当密度阈值 δ_0=0,1,2,… ,10 时，数据集中噪声数据的个数。以下是噪声数据对象的定义：

定义 9　噪声数据对象 x_i。噪声数据对象 x_i 符合以下两个条件：x_i 所在的网格单元为稀疏的网格单元；x_i 所在的网格单元不与稠密的网格单元相邻接。

只有符合以上两个条件的数据对象才是噪声数据。在不同的密度阈值 δ_0 下，将获得不同数量的噪声数据。以密度阈值 δ_0 为横坐标，噪声个数 *noise* 为纵坐标作二维坐标系，将得到的这些数据画在坐标系中，再用平滑的曲线来拟合这些数据的趋势，所得的曲线即为噪声曲线。如图 3 所示，为数据集 dataset1 在图 1 的划分下的噪声曲线图。

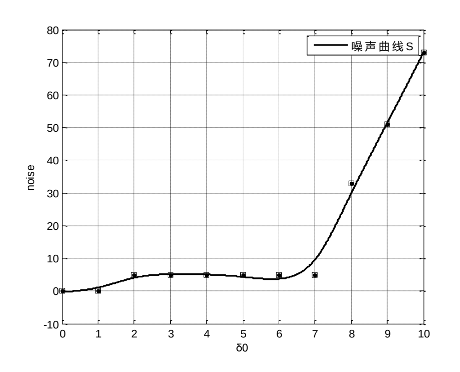


图 3（dataset1 的噪声曲线）

图 3 是数据集 dataset1 在图 1 的划分下的噪声曲线图。从图 3 可以看出，曲线在[0，7]内比较平缓，所对应的噪声数据 *noise* 也相对较小。当 δ_0 的取值从 7 增长到 8 时，曲线发生了急剧的变化。经分析，曲线急剧变化的原因是当密度阈值 δ_0=8 时，有一个密度较小的类簇被误认为是噪声数据，因此导致了噪声数量的急剧增长。所以，对于 dataset1 中的数据对象，密度阈值设置为 7 是比较合理的。

2.3　类簇边缘的处理

网格密度聚类算法聚类精度不高的主要原因之一是类簇边缘的部分数据被划分到稀疏的单元格中，当做噪声数据而被丢弃，这也是聚类结果中类簇边缘失去平滑度的直接原因。为了提高聚类的精度，必须对类簇的边缘数据进行提取，而不是简单的将其当做噪声数据丢弃。

为叙述方便，进行以下定义：

定义 10　网格单元 $grid_i$ 和 $grid_j$ 的相似性 *Sim*。$grid_i$ 和 $grid_j$ 的相似性 *Sim* 是指 $grid_i$ 中数据对象间的平均欧式距离和 $grid_j$ 中数据对象间的平均欧式距离的比值：

$$Sim(grid_i, grid_j) = \chi\left(\frac{dis_means(grid_i)}{dis_means(grid_j)}\right) \quad (7)$$

其中：$dis_means(grid_k)$是指网格单元 $grid_k$ 中数据对象间的平均欧氏距离。函数 $\chi(x)$的定义如下：

$$\chi(x) = \begin{cases} x & x \le 1 \\ 1/x & x > 1 \end{cases} \quad (8)$$

由表达式可知，$Sim(grid_i,grid_j)$的取值范围是(0,1]。*Sim* 的

值越大，两个网格单元的相似度越大。

定义 11　网格单元 $grid_i$ 和 $grid_j$ 的吸引性 *Attra*。$grid_i$ 和 $grid_j$ 的吸引性 *Attra* 是指 $grid_i$ 中所有数据的均值与 $grid_j$ 中所有数据的均值之间的欧式距离的倒数，形式化定义如式（9）所示。

$$Attra\left(grid_i, grid_j\right) = \frac{1}{dis\left(means\left(grid_i\right), means\left(grid_j\right)\right)} \quad (9)$$

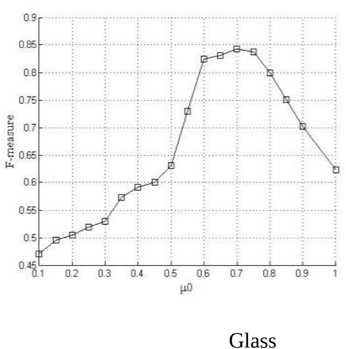
其中：$means(grid_k)$是网格 $grid_k$ 中所有数据对象的均值，$dis(x_i,x_j)$是数据对象 x_i 和 x_j 之间的欧式距离。

对于和稠密网格单元 $grid_m$ 接壤（有一个公共边或者公共面，高维情况下是超面）的稀疏网格单元格 $grid_n$，分别计算 $grid_m$ 和 $grid_n$ 之间的相似性和吸引性这两个指标，当这两个指标满足式（10）（11）的关系时，就将这个稀疏的网格单元中的数据对象合并到稠密网格单元数据所在的类簇中。

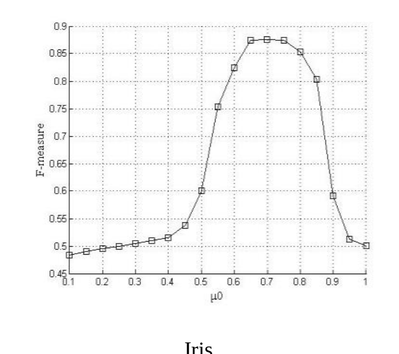
$$Sim\left(grid_m, grid_n\right) \geq \mu_0 \quad (10)$$

$$Attra\left(grid_m, grid_n\right) \geq \frac{1}{grid_wide} \quad (11)$$

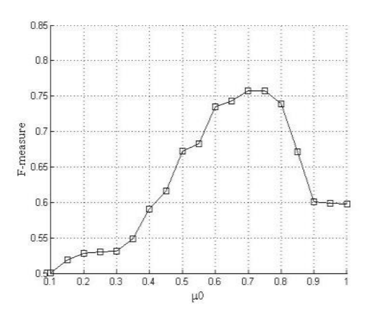
式（11）中的 $grid_wide$ 是 $grid_m$ 和 $grid_n$ 相异维的网格宽度。式（10）中，μ_0 是相似性阈值(一个常数)，不同的 μ_0 值将对聚类的精度有一定的的影响。为了选取合适的 μ_0 值，本文选取了 UCI 中三个真实高维数据集进行实验，数据集的详细信息如表 1 所示。对于不同的 μ_0 值（范围是[0.1,1]），分别运用本文的算法对三种数据集进行聚类操作，其对应的 F-measure 值的变化如图 4 所示。其中，F-measure 值越大，说明聚类效果越好，反之越差。



Glass



Iris



Wine

图 4　(不同 μ0 值所对应的 F-measure 值)

表 1　UCI 高维数据集信息

数据集名称	数据集大小	属性类型	属性个数
Glass	214	数值型	10
Iris	150	数值型	4
Wine	178	数值型	13

从图 4 中曲线的变化可知，随着相似性阈值 μ_0 值的变化，F-measure 值刚开始呈递增趋势，这说明随着 μ_0 值的增加，聚类的结果逐渐的改善。当 μ_0 值增加的一定程度时，F-measure 的值开始下降，说明 μ_0 值增大到一定程度，聚类的结果会逐渐变差。从三个曲线中可以看出，μ_0 的值在区间[0.60,0.80]时 F-measure 的值较大，因此 μ_0 的值取 0.60 到 0.80 中的某个值较为合适。因此本文取 μ_0=0.70。

至此，SA-BGD 算法已经对网格密度聚类算法中存在的三个缺陷进行了修改。本文接下来的安排如下：第三节将对改进后的算法进行仿真实验；第四节对全文进行总结。

3　仿真实验及结果分析

SA-BGD 算法首先利用数据集的分布情况计算出维度分割参数 m 的值，然后利用噪声曲线获得合理的密度阈值 δ_0，最后还增加了对类簇边缘的处理技术，从而改进了基于网格密度聚类算法存在的三个缺陷。

3.1　SA-BGD 算法的执行流程

SA-BGD 算法的执行流程如下：

a)输入待聚类数据集 S。

b)根据输入的数据集 S 和公式(6)计算出维度分割参数 m。

c)根据维度分割参数 m，分割数据空间的每一维，并根据式（1）将集合 S 中的数据对象映射到相应的网格单元中。

d)根据不同的密度阈值 δ_0 检测数据集中噪声数据的数量，绘制噪声曲线，从噪声曲线中获得最佳的密度阈值 δ_0_best。

e)根据最佳密度阈值 δ_0_best 识别稀疏和稠密网格单元。

f)从网格单元集合 G 中选择一个未被标记的稠密的网格单元，寻找与该网格单元相连的稠密网格单元的最大连通区域，并将这些网格单元标记为 *visited*（已访问）。

g)重复 f)，直到集合 G 中不存在未被标记的稠密网格单元为止。

h)对集合 G 中稀疏的网格单元进行边缘检测，将符合式（10）（11）的稀疏网格单元加入到与它接壤的稠密网格单元所在的最大连通区域中。

i)每一个最大连通区域就是一个类簇，输出这些连通区域中所有的数据对象

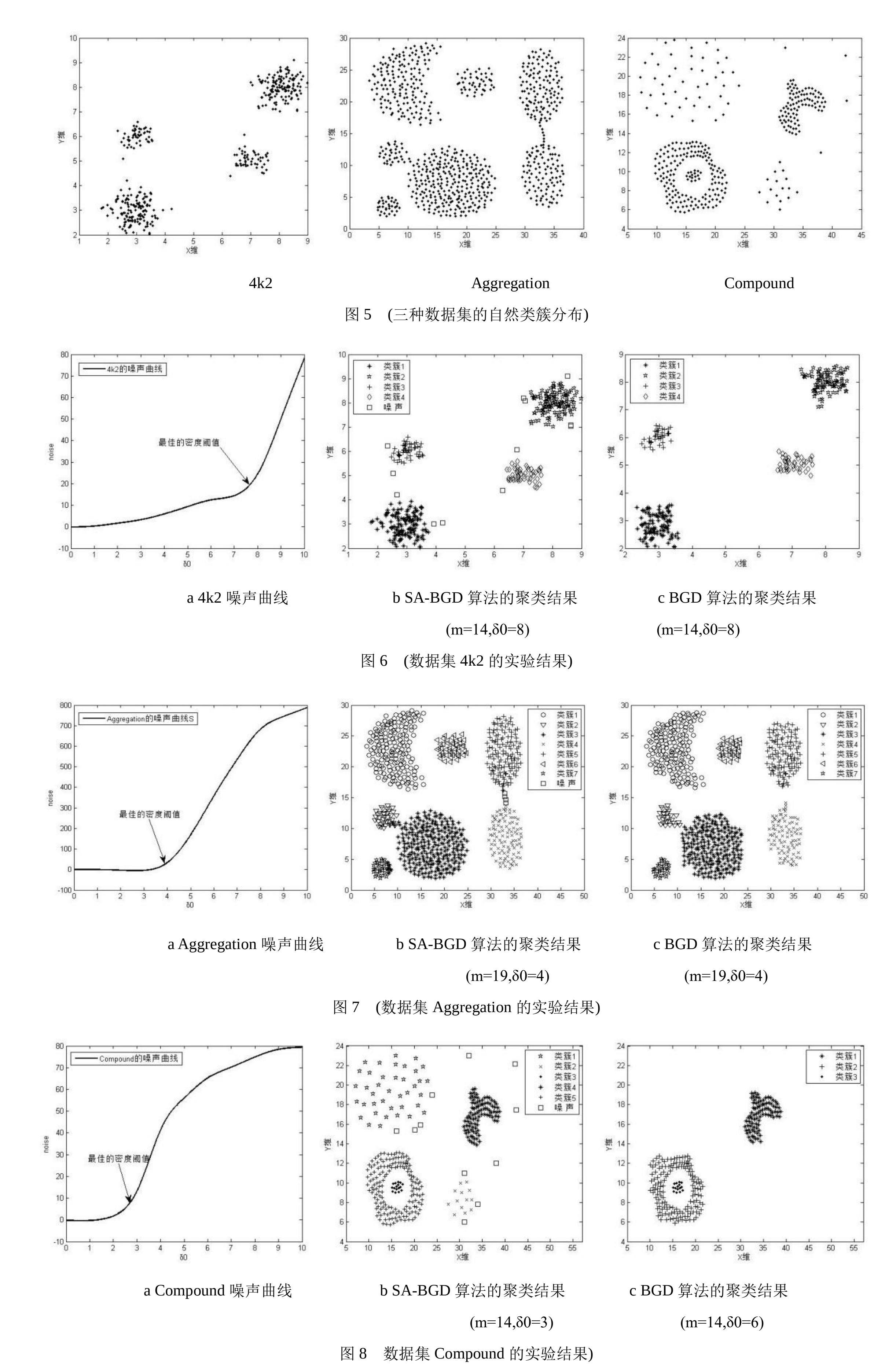
3.2　仿真实验及结果分析

3.2.1 二维数据集的实验

本节对 SA-BGD 算法和 BGD（网格密度聚类算法）进行实验比较，为了便于观察实验效果，实验数据采用了基于网格密度聚类算法中最常用的二维数据集。数据集的自然类簇分布

情况如图 5 所示，数据集 4k2、Aggregation、Compound 的真实类簇数目分别为 4、7、5。并且三种数据集都有一定的噪声数据。

在三种数据集上，分别用 SA-BGD 算法和 BGD 算法进行聚类操作，三种数据集的噪声曲线和聚类结果如图 6~8 所示。



4k2　　Aggregation　　Compound

图 5　(三种数据集的自然类簇分布)

a 4k2 噪声曲线　　b SA-BGD 算法的聚类结果 (m=14,δ0=8)　　c BGD 算法的聚类结果 (m=14,δ0=8)

图 6　(数据集 4k2 的实验结果)

a Aggregation 噪声曲线　　b SA-BGD 算法的聚类结果 (m=19,δ0=4)　　c BGD 算法的聚类结果 (m=19,δ0=4)

图 7　(数据集 Aggregation 的实验结果)

a Compound 噪声曲线　　b SA-BGD 算法的聚类结果 (m=14,δ0=3)　　c BGD 算法的聚类结果 (m=14,δ0=6)

图 8　数据集 Compound 的实验结果)

从图 6（a）中 4k2 的噪声曲线可以看出，数据集 4k2 的最佳密度阈值是 8，运用由式（6）计算而来的维度分割参数 m=14 和最佳密度阈值 δ_0=8 进行聚类操作，SA-BGD 算法和 BGD 算法的聚类结果如图 6（b）（c）所示，BGD 算法的类簇边缘呈直线型，丢失了类簇边缘的部分数据对象，SA-BGD 算法的聚类结果要优于 BGD 算法。从图 7（a）中的噪声曲线中可以看出，数据集 Aggregation 的最佳密度阈值是 4，运用由式（6）计算出来的分割参数 m=19 和密度阈值 δ_0=4 进行聚类操作的结果如图 7（b）（c）所示。图 7（b）是 SA-BGD 算法的聚类结果，图 7（c）是 BGD 算法的聚类结果，与图 7（c）中的聚类结果相比较，图 7（b）的类簇边缘平滑了不少，这就有效的提高了聚类的质量。图 8（a）中，从数据集 Compound 的噪声曲线可以看出，数据集 Compound 的最佳密度阈值 δ_0=3，式（6）计算的维度分割参数 m=14，运用这两个数值作为输入参数，SA-BGD 算法的聚类结果如图 8b 所示，从结果来看，SA-BGD 聚类算法可以很好的显示出密度比较稀疏的类簇。图 8（c）是用 BGD 算法在数据集 Compound 上进行的聚类操作，其中的输入参数为 m=14，δ_0=6，聚类结果是很差的，首先它漏掉了两个密度较小的类簇，致使聚类结果严重失真；其次，由于原算法没有对类簇边缘的稀疏网格单元进行处理，导致类簇边缘失去原有的平滑，聚类的精度不高。

3.2.2 高维数据集的实验

为了验证 SA-BGD 算法的鲁棒性，本小节采用 SA-BGD 算法对表 1 中的 UCI 真实的高维数据集进行聚类操作，并且将其聚类结果与 BGD 算法、K-means 算法、BIRCH 算法、OPTICS 算法的聚类结果进行比较。

高维数据的聚类结果无法采用直观的图片来展示，为此本文采用聚类结果常用的评价指标 ARI（adjusted Rand index 兰德指数）、F-measure 及 E(c)（误差平方和）来衡量四中聚类算法在上述 3 中数据集上面的聚类结果的优劣。其聚类结果对比如表 2 所示。

三个指标中，E(c)越小，表示聚类质量越高，反之较差；ARI 和 F-measure 的值越小，代表聚类质量越高，反之越差。从表 2 可以看出，除去数据集 Wine，SA-BGD 算法均可取得优于 K-means、BIRCH、OPTICS 及 BGD 算法的结果。在 Wine 数据集上面，虽然 SA-BGD 算法的聚类质量并没有提升，但也基本和其它算法类似。SA-BGD 算法的聚类结果较好的原因是因为根据数据集的本身分布进行了统计，从而计算出了分割宽度，然后有根据噪声曲线选择除了较合适的密度阈值，而且增加了边缘提取技术，从而提改了聚类的精度。

表 2　在三种数据集上五种算法的聚类结果

算法	数据	E(c)	ARI	F-measure
	Glass	225.10	0.343	0.749
K-means	Iris	102.78	0.456	0.652
	Wine	16913.26	0.715	0.776
BIRCH	Glass	228.10	0.331	0.645
	Iris	110.34	0.219	0.543
	Wine	16923.54	0.456	0.603
	Glass	223.15	0.782	0.732
OPTICS	Iris	101.74	0.542	0.801
	Wine	16910.45	0.783	0.792
	Glass	230.19	0.123	0.623
BGD	Iris	112.57	0.113	0.501
	Wine	16949.41	0.102	0.598
	Glass	222.47	0.891	0.843
SA-BGD	Iris	100.93	0.673	0.876
	Wine	16914.03	0.704	0.757

4　结束语

本文针对网格密度聚类算法存在的缺陷，提出了参数自适应的网格密度聚类算法。运用数据集的分布信息，自适应的计算维度分割参数 m 的值，并且运用噪声曲线来获取网格单元的最佳密度阈值 δ_0_best，还对聚类结果进行了边缘处理技术，提高了聚类结果的精度。仿真实验显示，参数自适应的网格密度聚类算法能取得更好的聚类效果。但是，SA-BGD 算法需要人工的从噪声曲线中识别最佳的密度阈值，这就使得算法需要人工干预。如何从噪声曲线中自动的获取最佳密度阈值，这是下一步将要重点解决的问题之一。

参考文献：

[1] 张敏，于剑. 基于划分的模糊聚类算法 [J]. 软件学报，2004, 15 (6): 858-868. (Zhang Min, Yu Jian. Fuzzy partitional clustering algorithms [J]. Journal of Software, 2004, 15 (6): 858-868.)

[2] 黄韬，刘胜辉，谭艳娜. 基于 K-means 聚类算法的研究 [J]. 计算机技术与发展，2011, 21 (7): 54-57, 62. (Huang Tao, Liu Shenghui, Tan Yanna. Research of clustering algorithm based on K-means [J]. Computer Technology & Development, 2011, 21 (7): 54-57, 62.)

[3] 夏宁霞，苏一丹，覃希. 一种高效的 K-medoids 聚类算法 [J]. 计算机应用研究，2010, 27 (12): 4517-4519. (Xia Ningxia, Su Yidan, Qin Xi. Efficient K-medoids clustering algorithm [J]. Application Research of Computers, 2010, 27 (12): 4517-4519.)

[4] Prakash J, Singh P K. An effective multiobjective approach for hard partitional clustering [J]. Memetic Computing, 2015, 7 (2): 93-104.

[5] 韩忠明，陈妮，张慧，等. 一种非对称距离下的层次聚类算法 [J]. 模式识别与人工智能，2014, 27 (5): 410-416. (Han Zhongming, Chen Ni, Zhang Hui, *et al.* A hierarchical clustering algorithm based on asymmetric distance [J]. Pattern Recognition & Artificial Intelligence, 2014, 27 (5): 410-416.)

[6] Carlsson G, Mémoli F, Ribeiro A, *et al.* Hierarchical clustering of asymmetric networks [C]// Advances in Data Analysis & Classification. 2016, 12: 1-41.

[7] Aliahmadipour L, Eslami E. GHFHC: generalized hesitant fuzzy

hierarchical clustering algorithm [J]. International Journal of Intelligent Systems, 2016, 31 (9): 855-871.

[8] 伍恒，李文杰，蒋旻．引入信息熵的 CURE 聚类算法 [J]. 计算机应用研究, 2017, 34 (8): 2303-2305. (Wu Heng, LI Wenjie, Jiang Min. Modified CURE clustering alogrithm based on entropy [J]. Application Research Of Computers, 2017, 34 (8): 2303-2305.)

[9] 夏鲁宁，荆继武．SA-DBSCAN: 一种自适应基于密度聚类算法 [J]. 中国科学院研究生院学报, 2009, 26 (4): 530-538 (Xia Luning, Jing jiwu. SA-DBSCAN: A self-adaptive density-based clustering algorithm [J]. Journal of the Graduate School of the Chinese Academy of Sciences, 2009, 26 (4): 530-538.)

[10] 武佳薇，李雄飞，孙涛，等．邻域平衡密度聚类算法 [J]. 计算机研究与发展, 2010, 47 (6): 1044-1052. (Wu Jiawei, Li Xiongfei, Sun Tao, *et al.* A density-based clustering algorithm concerning neighborhood balance [J]. Journal of Computer Research & Development, 2010, 47 (6): 1044-1052.)

[11] Khalid S, Razzaq S. TOBAE: A Density-based Agglomerative Clustering Algorithm [M]. New York: Springer-Verlag, 2015.

[12] 赵慧，刘希玉，崔海青．网格聚类算法 [J]. 计算机技术与发展，2010, 20 (09): 83-85, 89. (Zhao Hui, Liu Xiyu, Cui Haiqing. Grid-based clustering algorithm [J]. Computer Technology & Development, 2010, 20 (9): 83-85, 89.)

[13] Chen Ning, Chen An, Zhou Longxiang. An Incremental grid density-based clustering algorithm [J]. Journal of Software, 2002 (01): 1-7.

[14] 杨洁，王国胤，王飞．基于密度峰值的网格聚类算法 [J]. 计算机应用，2017, 37 (11): 3080-3084. (Yang Jie, Wang Guoyin, Wang Fei. Grid clustering algorithm based on density peaks [J]. Journal of Computer Applications, 2017, 37 (11): 3080-3084.)

[15] 宋浩远．基于模型的聚类方法研究 [J]. 重庆科技学院学报：自然科学版, 2008 (3): 71-73. (Song Haoyuan. Study on Model-based Clustering Methods [J]. Journal of Chongqing University of Science & Technology, 2008 (3): 71-73.)

[16] Tang Yang, Browne R P, Mcnicholas P D. Model based clustering of high-dimensional binary data [J]. Computational Statistics & Data Analysis, 2015, 87: 84-101.

[17] 胡泱，陈刚．一种有效的基于网格和密度的聚类分析算法 [J]. 计算机应用, 2003, 23 (12): 64-67. (Hu Yang, Chen Gang. An effective cluster analysis algorithm based on grid and intensity [J]. Journal of Computer Applications, 2003, 23 (12): 64-67.)

[18] Triplet T. PatchWork, a scalable density-grid clustering algorithm [C]// Proc of ACM Symposium on Applied Computing. New York: ACM Press, 2016: 824-831.

[19] Meila M, Shi Jianguo. A random walks view of spectral segmentation [J]. Ai & Statistics, 2001.

[20] Hofmann T, Buhmann J M. Pairwise Data Clustering by Deterministic Annealing [J]. IEEE Trans on Pattern Analysis & Machine Intelligence, 2002, 19 (1): 1-14.

[21] 于佐军，秦欢．基于改进蜂群算法的 K-means 算法 [J]. 控制与决策, 2018, 33 (1): 181-185. (Yu Zuojun, Qin Huan. K-means algorithm based on improved artificial bee colony algorithm [J]. Control and Decision, 2018, 33 (1): 181-185.)